

Estructura básica de un programa en C

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Hola");
```

```
    return 0;
```

```
}
```

Estructura básica de un programa en C

```
#include <stdio.h>
```

```
int main()  
{  
    printf("Hola");  
    return 0;  
}
```

Directivas del Preprocesador:

- No son parte del código ejecutable
- Son “indicaciones” que se dan al proceso de compilación
- Las directivas **#include** importan la “documentación” de las funciones a utilizar, listadas en “archivos de cabecera” ***.h**
- También se utilizarán mucho las directivas **#define**

Estructura básica de un programa en C

```
#include <stdio.h>
```

```
int main()  
{  
    printf("Hola");  
    return 0;  
}
```

Función `main()` → Programa principal

- Debe existir en cualquier programa ejecutable
- Consta del encabezado `int main()`
- y del código ejecutable encerrado entre llaves `{...}`
- El código se ejecutará ordenadamente desde la llave superior `{` (inicio del algoritmo) hasta la llave inferior `}` (fin del algoritmo)

Estructura básica de un programa en C

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Hola");
```

```
    return 0;
```

```
}
```

Encabezado de la función main()

Estructura básica de un programa en C

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Hola");
```

```
    return 0;
```

```
}
```

código ejecutable

- Las sentencias terminan en ;
- Contienen expresiones, operaciones y llamadas a funciones
- En un código típico también habrán:
 - estructuras para el control de flujo
 - Declaración de variables

Estructura básica de un programa en C

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Hola");
```

```
    return 0;
```

```
}
```

Llamada a la función printf()

- La función **printf()** equivale a la instrucción **Escribir** del pseudocódigo
- Solo puede utilizarse si se incluye **stdio.h**

Estructura básica de un programa en C

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Hola");
```

```
    return 0;
```

```
}
```

Instrucción de retorno

- Su ejecución finaliza con el programa
- Debe ir al final del programa principal (antes de la llave de cierre }

Nivel	Operadores	Descripción	Asoci.
1	() [] -> .	Acceso a un elemento de un vector y paréntesis	Izquierdas
2	+ - ! ~ * & ++ -- (cast) sizeof	Signo (unario), negación lógica, negación bit a bit Acceso a un elemento (unarios): puntero y dirección Incremento y decremento (pre y post) Conversión de tipo (<i>casting</i>) y tamaño de un elemento	Derechas
3	* / %	Producto, división, módulo (resto)	Izquierdas
4	+ -	Suma y resta	Izquierdas
5	>> <<	Desplazamientos	Izquierdas
6	< <= >= >	Comparaciones de superioridad e inferioridad	Izquierdas
7	== !=	Comparaciones de igualdad	Izquierdas
8	&	Y (<i>And</i>) bit a bit (binario)	Izquierdas
9	^	O-exclusivo (<i>Exclusive-Or</i>) (binario)	Izquierdas
10		O (<i>Or</i>) bit a bit (binario)	Izquierdas
11	&&	Y (<i>And</i>) lógico	Izquierdas
12		O (<i>Or</i>) lógico	Izquierdas
13	?:	Condicional	Derechas
14	= *= /= %= += -= >>= <<= &= ^= =	Asignaciones	Derechas
15	,	Coma	Izquierdas

Operadores de C

Nivel	Operadores	Descripción	Asoci.
1	() [] -> .	Acceso a un elemento de un vector y paréntesis	Izquierdas
2	+ - ! ~ * & ++ -- (cast) sizeof	Signo (unario), negación lógica, negación bit a bit Acceso a un elemento (unarios): puntero y dirección Incremento y decremento (pre y post) Conversión de tipo (<i>casting</i>) y tamaño de un elemento	Derechas
3	* / %	Producto, división, módulo (resto)	Izquierdas
4	+ -	Suma y resta	Izquierdas
5	>> <<	Desplazamientos	Izquierdas
6	< <= >= >	Comparaciones de superioridad e inferioridad	Izquierdas
7	== !=	Comparaciones de igualdad	Izquierdas
8	&	Y (<i>And</i>) bit a bit (binario)	Izquierdas
9	^	O-exclusivo (<i>Exclusive-Or</i>) (binario)	Izquierdas
10		O (<i>Or</i>) bit a bit (binario)	Izquierdas
11	&&	Y (<i>And</i>) lógico	Izquierdas
12		O (<i>Or</i>) lógico	Izquierdas
13	?:	Condicional	Derechas
14	= *= /= %= += -= >>= <<= &= ^= =	Asignaciones	Derechas
15	,	Coma	Izquierdas

Operadores Unarios (un único operando)

Operadores Aritméticos

Operadores relacionales

Operadores lógicos (AND/OR)

Operadores de asignación

Ejemplo básico con variables

```
#include <stdio.h>

int main()
{
    /* Declaración de variables */
    int Npizzas;
    float harina,sal,aceite,levadura,azucar,agua;

    /* Algoritmo */
    printf("Ingrese la cantidad de pizzas: ");
    scanf("%d", &Npizzas);

    harina = Npizzas/5.0;
    sal = Npizzas/5.0 * 2;
    aceite = Npizzas/5.0 * 3;
    levadura = Npizzas/5.0 * 50;
```

```
    azucar = Npizzas/5.0;
    agua = Npizzas/5.0 * 700;

    printf("Para amasar %d pizzas necesita:\n", Npizzas);
    printf("%f Kg de harina 000\n", harina);
    printf("%f cucharadas de sal\n", sal);
    printf("%f cucharadas de aceite\n", aceite);
    printf("%f gramos de levadura\n", levadura);
    printf("%f cucharadas de azucar\n", azucar);
    printf("%f cc de agua\n", agua);
    return 0;
```

```
}
```

Ejemplo básico con variables

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
/* Declaración de variables */
```

```
int Npizzas;
```

```
float harina,sal,aceite,levadura,azucar;
```

```
/* Algoritmo */
```

```
printf("Ingrese la cantidad de pizzas: ");
```

```
scanf("%d", &Npizzas);
```

```
harina = Npizzas/5.0;
```

```
sal = Npizzas/5.0 * 2;
```

```
aceite = Npizzas/5.0 * 3;
```

```
levadura = Npizzas/5.0 * 50;
```

```
azucar;
```

```
agua;
```

```
printf
```

```
printf("%f cucharadas de aceite\n", aceite);
```

```
printf("%f gramos de levadura\n", levadura);
```

```
printf("%f cucharadas de azucar\n", azucar);
```

```
printf("%f cc de agua\n", agua);
```

```
return 0;
```

```
}
```

Comentarios

- Es tratado como texto y no se compila
- encerrados entre /* y */
- Pueden tener varias líneas

Ejemplo básico con variables

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    /* Declaración de variables */
```

```
    int Npizzas;
```

```
    float harina,sal,aceite,levadura,azucar,agua;
```

```
    /* Algoritmo */
```

```
    printf("Ingrese la cantidad de pizzas: ");
```

```
    scanf("%d", &Npizzas);
```

```
    harina = Npizzas/5.0;
```

```
    sal = Npizzas/5.0 * 2;
```

```
    aceite = Npizzas/5.0 * 3;
```

```
    levadura = Npizzas/5.0 * 50;
```

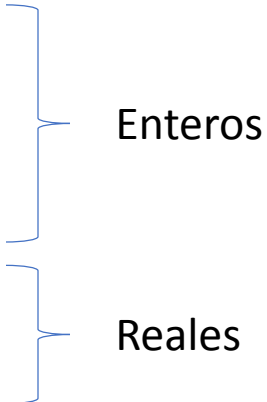
```
}
```

Declaración de variables

- Se deben declarar **TODAS** las variables antes de cualquier sentencia ejecutable
- El formato es:
 <modificadores de tipo> <tipo> <nombre_variable>
- El identificador de la variable puede contener letras, números y guiones bajos (_)
- Solo puede comenzar con letras o guión bajo

Tipos de variables

Los tipos básicos son;

- **char** (caracter o también entero de 8 bits)
 - **short** (entero corto) (2 bytes)
 - **int** (entero) (2 o 4 bytes) (4 en nuestro caso)
 - **long** (entero grande) (4 bytes)
- 
- Enteros
- **float** (punto flotante simple precisión)
 - **double** (punto flotante doble precisión)
- Reales

Podemos usar el modificador **unsigned** en la declaración de char, short, int y long, para declararlo como variable entera **sin signo**

Tipos de variables

Los tipos básicos son;

- **char** (caracter o también entero de 8 bits)
- **short** (entero corto) (2 bytes)
- **int** (entero) (2 o 4 bytes) (4 en nuestro caso)
- **long** (entero grande) (4 bytes)
- **float** (punto flotante simple precisión)
- **double** (punto flotante doble precisión)

El tipo de dato char, si bien guarde un número entero, se utiliza para representar caracteres según la tabla ASCII

Enteros

Reales

Podemos usar el modificador **unsigned** en la declaración de char, short, int y long, para declararlo como variable entera **sin signo**

Tipos de variables

char	−128 to 127
unsigned char	0 to 255
short int	−32,768 to 32,767
unsigned short int	0 to 65,535
int	−2,147,483,648 to 2,147,483,647
unsigned int	0 to 4,294,967,295
long int	−2,147,483,648 to 2,147,483,647
unsigned long int	0 to 4,294,967,295
float	1.17×10^{-38} to 3.40×10^{38}
double	2.22×10^{-308} to 1.79×10^{308}

Caracteres ASCII de control

00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc.vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sínc)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

Caracteres ASCII imprimibles

32	espacio	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	\$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(72	H	104	h
41)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[123	{
60	<	92	\	124	
61	=	93]	125	}
62	>	94	^	126	~
63	?	95	_		

ASCII extendido (Página de código 437)

128	Ç	160	á	192	Ł	224	Ó
129	ü	161	í	193	ł	225	õ
130	é	162	ó	194	Ł	226	ô
131	â	163	ú	195	ł	227	ò
132	ä	164	ñ	196	—	228	ö
133	à	165	Ñ	197	†	229	õ
134	â	166	ª	198	ã	230	μ
135	ç	167	º	199	Ä	231	þ
136	ê	168	¿	200	Ł	232	ƒ
137	ë	169	®	201	ƒ	233	ú
138	è	170	¬	202	Ł	234	û
139	ï	171	½	203	ƒ	235	ù
140	î	172	¼	204	ƒ	236	ý
141	ì	173	¡	205	=	237	ÿ
142	Ä	174	«	206	ƒ	238	—
143	Å	175	»	207	□	239	·
144	É	176	⋮	208	ð	240	≡
145	æ	177	⋮	209	Ð	241	±
146	Æ	178	⋮	210	Ê	242	—
147	ô	179		211	Ë	243	¾
148	ö	180	†	212	Ê	244	¶
149	ò	181	À	213	Ì	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	º
152	ÿ	184	©	216	Ï	248	°
153	Ö	185	¶	217	Ɔ	249	ˆ
154	Ü	186	¶	218	Ɔ	250	˙
155	ø	187	¶	219	■	251	˚
156	£	188	¶	220	■	252	˚
157	Ø	189	¢	221	⋮	253	˚
158	×	190	¥	222	⋮	254	■
159	f	191	¬	223	■	255	nbsp

Caracteres ASCII de control

00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc.vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sínc)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

Caracteres ASCII imprimibles

32	espacio	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	\$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(72	H	104	h
41)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[123	{
60	<	92	\	124	
61	=	93]	125	}
62	>	94	^	126	~
63	?	95	_		

ASCII extendido (Página de código 437)

128	Ç	160	á	192	Ł	224	Ó
129	ü	161	í	193	ł	225	ô
130	é	162	ó	194	Ł	226	ò
131	â	163	ú	195	ł	227	õ
132	ä	164	ñ	196	—	228	ö
133	à	165	Ñ	197	†	229	ő
134	â	166	ä	198	‡	230	û
135	ä	167	å	199	§	231	ü
136	å	168	æ	200	¶	232	ý
137	æ	169	ç	201	§	233	ÿ
138	ç	170	È	202	¶	234	ÿ
139	è	171	É	203	¶	235	ÿ
140	é	172	Ê	204	¶	236	ÿ
141	ê	173	Ë	205	¶	237	ÿ
142	ë	174	Ì	206	¶	238	ÿ
143	ì	175	Í	207	¶	239	ÿ
144	É	176	Ï	208	¶	240	ÿ
145	æ	177	Ð	209	¶	241	±
146	Æ	178	Ñ	210	¶	242	—
147	ô	179	Ò	211	¶	243	¾
148	ö	180	Ó	212	¶	244	¶
149	ò	181	Ô	213	¶	245	§
150	û	182	Å	214	¶	246	÷
151	ù	183	À	215	¶	247	°
152	ÿ	184	©	216	¶	248	°
153	Ö	185	ª	217	¶	249	°
154	Ü	186	»	218	¶	250	°
155	ø	187	«	219	¶	251	°
156	£	188	»	220	¶	252	°
157	Ø	189	¢	221	¶	253	°
158	×	190	¥	222	¶	254	■
159	f	191	¬	223	¶	255	nbsp

Caracteres numéricos

- Desde el carácter '0' al carácter '9' ordenados en forma creciente

Caracteres ASCII de control

00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc.vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sínc)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

Caracteres ASCII imprimibles

32	espacio	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	\$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(72	H	104	h
41)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[123	{
60	<	92	\	124	
61	=	93]	125	}
62	>	94	^	126	~
63	?	95	_		

ASCII extendido (Página de código 437)

128	Ç	160	á	192	Ł	224	Ó
139	ĩ	171	½	203	ƒ	235	Ů
140	î	172	¼	204	ƒ	236	ý
141	ï	173	ı	205	=	237	Ÿ
142	Ä	174	«	206	ƒ	238	—
143	Å	175	»	207	□	239	˙
144	É	176	⋮	208	ø	240	≡
145	æ	177	⋮	209	Ð	241	±
146	Æ	178	⋮	210	Ê	242	—
147	ô	179		211	Ë	243	¾
148	ö	180	†	212	È	244	¶
149	ò	181	À	213	Ì	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	˚
152	ÿ	184	©	216	Ï	248	°
153	Ö	185	ƒ	217	Ɔ	249	ˆ
154	Ü	186		218	ƒ	250	˙
155	ø	187	ƒ	219	■	251	˙
156	£	188	ƒ	220	■	252	˙
157	Ø	189	¢	221	˙	253	˙
158	×	190	¥	222	˙	254	■
159	f	191	ƒ	223	■	255	nbsp

Caracteres alfabéticos mayúscula

- Desde el carácter 'A' al carácter 'Z' ordenados en forma creciente
- No se incluye a la 'Ñ' ni a las vocales acentuadas (están en la tabla extendida)

Caracteres ASCII de control

00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc.vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sínc)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

Caracteres ASCII imprimibles

32	espacio	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	\$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(72	H	104	h
41)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[123	{
60	<	92	\	124	
61	=	93]	125	}
62	>	94	^	126	~
63	?	95	_		

ASCII extendido (Página de código 437)

128	Ç	160	á	192	Ł	224	Ó
139	ĩ	171	½	203	ƒ	235	Ů
140	î	172	¼	204	ƒ	236	ý
141	ï	173	ı	205	=	237	Ÿ
142	Ä	174	«	206	ƒ	238	—
143	Å	175	»	207	□	239	˙
144	É	176	⋮	208	ø	240	≡
145	æ	177	⋮	209	Ð	241	±
146	Æ	178	⋮	210	Ê	242	—
147	ô	179		211	Ë	243	¾
148	ö	180	†	212	È	244	¶
149	ò	181	À	213	Ì	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	˚
152	ÿ	184	©	216	Ï	248	°
153	Ö	185	ƒ	217	Ɔ	249	ˆ
154	Ü	186		218	ƒ	250	˙
155	ø	187	ƒ	219	■	251	˙
156	£	188	ƒ	220	■	252	˙
157	Ø	189	¢	221	˙	253	˙
158	×	190	¥	222	˙	254	■
159	f	191	ƒ	223	■	255	nbsp

Caracteres alfabéticos minúscula

- Desde el carácter 'a' al carácter 'z' ordenados en forma creciente
- No se incluye a la 'ñ' ni a las vocales acentuadas (están en la tabla extendida)

Expresiones con distintos tipos

Conversión de tipos de datos: *Implícita y Explícita*

Implícita:

La conversión de tipo implícita se da cuando se mezclan operandos de tipos distintos en una misma expresión. Esta conversión se realiza aplicando las siguientes reglas en el orden en que están escritas:

- si un operando es tipo 'long double', el otro se convierte a ese tipo.
- si un operando es tipo 'double', el otro se convierte a ese tipo.
- si un operando es tipo 'float', el otro se convierte a ese tipo.
- si un operando es tipo 'char' ó 'unsigned char', se convierte a 'int'.
- si un operando es tipo 'unsigned long', el otro se convierte a ese tipo.
- si un operando es tipo 'long', el otro se convierte a ese tipo.
- si un operando es tipo 'unsigned int', el otro se convierte a ese tipo.
- en otro caso, ambos son 'int'.

Explícita:

Indicando el tipo entre paréntesis.

Ejemplo:

```
int num = (int) (12.25/3);
```

Expresiones con distintos tipos

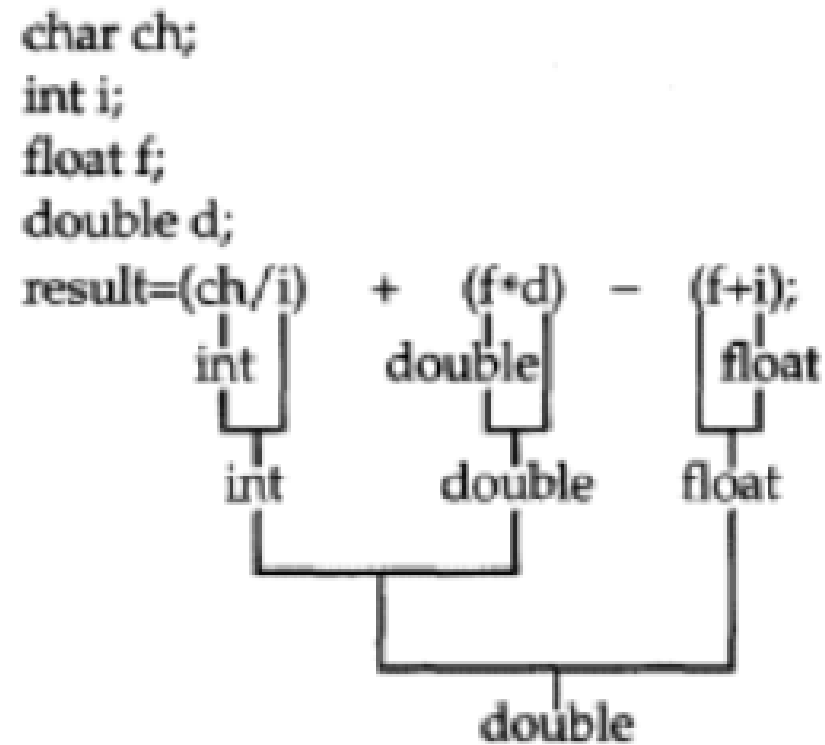


Figure 2-2
A type conversion example

Ejemplo básico con variables

```
#include <stdio.h>

int main()
{
    /* Declaración de variables */
    int Npizzas;
    float harina,sal,aceite,levadura,azucar,agua;

    /* Algoritmo */
    printf("Ingrese la cantidad de pizzas: ");
    scanf("%d", &Npizzas);

    harina = Npizzas/5.0;
    sal = Npizzas/5.0 * 2;
    aceite = Npizzas/5.0 * 3;
    levadura = Npizzas/5.0 * 50;
```

```
azucar = Npizzas/5.0;
agua = Npizzas/5.0 * 700;
```

```
printf("Para amasar %d pizzas necesita:\n", Npizzas);
printf("%f Kg de harina 000\n", harina);
```

Impresión de una cadena constante

```
printf("%f cucharadas de aceite\n", aceite);
printf("%f gramos de levadura\n", levadura);
printf("%f cucharadas de azucar\n", azucar);
printf("%f cc de agua\n", agua);

return 0;
```

```
}
```

Ejemplo básico con variables

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
/* Declaración de variables */
```

```
int Npizzas;
```

```
float harina;
```

Impresión de variables mezclada con texto

```
/* Algoritmo */
```

```
printf("Ingrese la cantidad de pizzas: ");
```

```
scanf("%d", &Npizzas);
```

```
harina = Npizzas/5.0;
```

```
sal = Npizzas/5.0 * 2;
```

```
aceite = Npizzas/5.0 * 3;
```

```
levadura = Npizzas/5.0 * 50;
```

```
azucar = Npizzas/5.0;
```

```
agua = Npizzas/5.0 * 700;
```

```
printf("Para amasar %d pizzas necesita:\n", Npizzas);
```

```
printf("%f Kg de harina\n", harina);
```

```
printf("%f cucharadas de sal\n", sal);
```

```
printf("%f cucharadas de aceite\n", aceite);
```

```
printf("%f gramos de levadura\n", levadura);
```

```
printf("%f cucharadas de azucar\n", azucar);
```

```
printf("%f cc de agua\n", agua);
```

```
return 0;
```

```
}
```


Ejemplo básico con variables

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
/* Decla
```

```
int Npiz
```

```
float ha
```

```
/* Algor
```

```
printf("
```

```
scanf("%
```

```
harina =
```

```
sal = Np
```

```
aceite = Npizzas/5.0 * 5;
```

```
levadura = Npizzas/5.0 * 50;
```

Cadena de control

- Contiene el texto a mostrar en pantalla
- Donde se quiere mostrar el valor de una variable se intercalan

ESPECIFICADORES DE FORMATO

- El especificador de formato depende del tipo de la variable que se quiere mostrar y del formato

```
azucar = Npizzas/5.0;
```

```
agua = Npizzas/5.0 * 700;
```

```
printf("Para amasar %d pizzas necesita:\n", Npizzas);
```

```
printf("%f Kg de harina 000\n", harina);
```

```
printf("%f cucharadas de sal\n", sal);
```

```
printf("%f cucharadas de aceite\n", aceite);
```

```
printf("%f gramos de levadura\n", levadura);
```

```
printf("%f cucharadas de azucar\n", azucar);
```

```
printf("%f cc de agua\n", agua);
```

```
return 0;
```

```
}
```

Especificadores de formato

- Comienza con %
- Finaliza con un carácter “**indicador de tipo**” que depende del tipo del valor que se quiere mostrar y del formato
- Los más usados son:
 - %d: enteros
 - %f : reales (float)
 - %lf: reales (double)
 - %c: caracteres
 - %s: cadenas
- Poniendo un número entre el % y el indicador de tipo se reserva un “**Ancho de campo**” mínimo para la impresión y el valor queda alineado a derecha:
 - %5d
 - %20s
- Un punto y un número luego del ancho de campo indica la “**precisión**” con la que se va a mostrar un real
 - %5.2f
 - %.2lf

ables

```
azucar = Npizzas/5.0;
```

```
agua = Npizzas/5.0 * 700;
```

```
printf("Para amasar %d pizzas necesita:\n", Npizzas);
```

```
printf("%f Kg de harina 000\n", harina);
```

```
printf("%f cucharadas de sal\n", sal);
```

```
printf("%f cucharadas de aceite\n", aceite);
```

```
printf("%f gramos de levadura\n", levadura);
```

```
printf("%f cucharadas de azucar\n", azucar);
```

```
printf("%f cc de agua\n", agua);
```

```
return 0;
```

```
}
```

```
aceite = Npizzas/5.0 * 3;
```

```
levadura = Npizzas/5.0 * 50;
```

El modificador **type** del especificador de formato puede ser:

Código	Formato
% c	Carácter
% d	Enteros con signo
% i	Enteros con signo
% ld	Enteros con signo <i>long</i>
% e	Coma flotante, notación científica (e minúscula)
% E	Coma flotante, notación científica (E mayúscula)
% f	Coma flotante <i>float</i>
% lf	Coma flotante <i>double</i>
% Lf	Coma flotante <i>long double</i>
% g	Usa e o f, el más corto
% G	Usa E o f, el más corto
% o	Octal sin signo
% s	Cadena de caracteres
% u	Enteros decimales sin signo
% x	Hexadecimales sin signo (letras minúsculas:a - f)
% X	Hexadecimales sin signo (letras mayúsculas:A -F)
% p	Mostrar un puntero
% n	El argumento asociado será un puntero a entero al que se le asigna el número de caracteres escritos
% %%	Visualizar el signo %

Lista de parámetros

- Contienen variables o expresiones cuyo valor se quiere mostrar (separados por coma) |
- Debe haber un modificador de formato para cada parámetro de la lista

```
int Npizzas;  
float harina,sal,aceite,levadura,azucar,agua;
```

```
/* Algoritmo */
```

```
printf("Ingrese la cantidad de pizzas: ");  
scanf("%d", &Npizzas);
```

```
harina = Npizzas/5.0;  
sal = Npizzas/5.0 * 2;  
aceite = Npizzas/5.0 * 3;  
levadura = Npizzas/5.0 * 50;
```

ables

```
azucar = Npizzas/5.0;
```

```
    Npizzas/5.0 * 700;
```

```
printf("Para amasar %d pizzas necesita:\n", Npizzas);
```

```
printf("%f Kg de harina 000\n", harina);
```

```
printf("%f cucharadas de sal\n", sal);
```

```
printf("%f cucharadas de aceite\n", aceite);
```

```
printf("%f gramos de levadura\n", levadura);
```

```
printf("%f cucharadas de azucar\n", azucar);
```

```
printf("%f cc de agua\n", agua);
```

```
return 0;
```

```
}
```

Ejemplo básico con va

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    /* Declaración de variables */
```

```
    int Npizzas;
```

```
    float harina,sal,aceite,levadura,azucar,agua;
```

```
    /* Algoritmo */
```

```
    printf("Ingrese la cantidad de pizzas: ");
```

```
    scanf("%d", &Npizzas);
```

```
    harina = Npizzas/5.0;
```

```
    sal = Npizzas/5.0 * 2;
```

```
    aceite = Npizzas/5.0 * 3;
```

```
    levadura = Npizzas/5.0 * 50;
```

Lectura de datos por teclado

- Una función muy usada es **scanf()** que al igual que **printf()** requiere la inclusión de **stdio.h**
- Permite la lectura de distintos tipos de valores
- Si se invoca numerosas veces a **scanf()** conviene invocar a la función **fflush(stdin);** para evitar errores

```
    printf("Ingrese Kg de harina 000\n", harina);
```

```
    printf("%f cucharadas de sal\n", sal);
```

```
    printf("%f cucharadas de aceite\n", aceite);
```

```
    printf("%f gramos de levadura\n", levadura);
```

```
    printf("%f cucharadas de azucar\n", azucar);
```

```
    printf("%f cc de agua\n", agua);
```

```
    return 0;
```

```
}
```

Ejemplo básico con va

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
/* Declaración de variables */
```

```
int Npizzas;
```

```
float harina,sal,aceite,levadura,azucar,agua;
```

```
/* Algoritmo */
```

```
printf("Ingrese la cantidad de pizzas: ");
```

```
scanf("%d", &Npizzas);
```

```
harina = Npizzas/5.0;
```

```
sal = Npizzas/5.0 * 2;
```

```
aceite = Npizzas/5.0 * 3;
```

```
levadura = Npizzas/5.0 * 50;
```

Cadena de control

- Incluye el o los modificadores de formato
- Son un subconjunto de los utilizados por **printf()**

```
printf("%f Kg de harina 000\n", harina);
```

```
printf("%f cucharadas de sal\n", sal);
```

```
printf("%f cucharadas de aceite\n", aceite);
```

```
printf("%f gramos de levadura\n", levadura);
```

```
printf("%f cucharadas de azucar\n", azucar);
```

```
printf("%f cc de agua\n", agua);
```

```
return 0;
```

```
}
```

Código	Significado
% c	Lee un único carácter
% d	Lee un entero
% i	Lee un entero
% ld	Lee un entero de tipo <i>long</i>
% e	Lee un número en coma flotante
% f	Lee un número en coma flotante
% lf	Lee un número en coma flotante <i>double</i>
% Lf	Lee un número en coma flotante <i>long double</i>
% g	Lee un número en coma flotante
% o	Lee un número octal
% s	Lee una cadena
% x	Lee un número hexadecimal
% p	Lee un puntero
% n	Recibe un valor entero igual al número de caracteres leídos
% u	Lee un entero sin signo

Ejemplo básico con va

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
/* Declaración de variables */
```

```
int Npizzas;
```

```
float harina,sal,aceite,levadura,azucar,agua;
```

```
/* Algoritmo */
```

```
printf("Ingrese la cantidad de pizzas: ");
```

```
scanf("%d", &Npizzas);
```

```
harina = Npizzas/5.0;
```

```
sal = Npizzas/5.0 * 2;
```

```
aceite = Npizzas/5.0 * 3;
```

```
levadura = Npizzas/5.0 * 50;
```

Lista de parámetros

- Lista de variables donde se almacenarán los valores ingresados (separadas por coma)
- Deben ser precedidas por el operador &

```
printf("Ingrese Kg de harina 000\n", harina);
```

```
printf("%f cucharadas de sal\n", sal);
```

```
printf("%f cucharadas de aceite\n", aceite);
```

```
printf("%f gramos de levadura\n", levadura);
```

```
printf("%f cucharadas de azucar\n", azucar);
```

```
printf("%f cc de agua\n", agua);
```

```
return 0;
```

```
}
```